

A blue parallelogram and a yellow parallelogram are positioned on the left side of the slide, overlapping each other and the dark blue background. The blue shape is on the left, and the yellow shape is to its right.

[saschavanessen@gmail.com](mailto:saschavanessen@gmail.com)

# Python Course

Week 4:  
Dictionaries and tuples



# Workshop Overview

1. Writing your first program
2. Making choices and reuse code
3. Loops and strings
4. Files and lists
5. Dictionaries and tuples

Acknowledgments: Structure of the workshop follows the book "Python for informatics" by Charles Severance. Several examples are from this book or the accompanying slides.



# Assignments last week

- Hobbies
- Truth
- Guess game 3



# Guess game: binary search

User thinks of number 9

					↓		↓	↓	↓	
1	2	3	4	5	6	7	8	9	10	
					higher		higher	higher	FOUND	

Keep dividing the space to look in half

# Dictionaries






# What is a dictionary?

- Key-value pairs
  - een -> one
  - twee -> two
  - drie -> three
- Curly brackets { }
- No saved order!

```
>>> NL2EN = dict()  
>>> print(NL2EN)
```



# Put data in dictionary

```
>>> NL2EN = dict()
```

```
>>> NL2EN["een"] = "one"
```

```
>>> NL2EN["twee"] = "two"
```

```
>>> NL2EN["drie"] = "three"
```

```
>>> print(NL2EN)
```

```
>>> altdict = {'een': 'one', 'twee': 'two', 'drie': 'three'}
```

```
>>> print(altdict)
```



# Get data out of dictionary

```
>>> print(NL2EN["twee"]) # shows value
>>> print(NL2EN["hallo"]) # does not exist: error

>>> for key in NL2EN:           # enter
...     print(key, NL2EN[key])  # tab
...                             # enter (close loop)
```





# Functions and operations

```
>>> len(NL2EN)    # amount of key-value pairs
```

```
>>> 'een' in NL2EN # True
```

```
>>> 'one' in NL2EN    # False
```

```
>>> 'one' in NL2EN.values()    # True
```

<https://docs.python.org/3.6/library/stdtypes.html#typesmapping>



# Count words

```
import sys

myfile = "plato.txt"

# Open file
try:
    fh = open(myfile, "r")
except:
    sys.exit("cannot open file")
```



# Count words

```
# Go through file, get the words and count the words
d = dict()
for line in fh:
    line = line.rstrip()    # remove newline
    line = line.lower()     # make all lowercase
    words = line.split()    # split the line on
                           # tab or space
    for word in words:
        d[word] = d.get(word, 0) + 1
```



# Count words

```
print("Show 10 (random) words and their frequency")
i = 0
for key in d:
    if i >= 10:
        break

    print(key, d[key])
    i += 1
```



# Count words (advanced)

```
# Advanced: print the 10 most used words
print("Show the TOP 10 words and their frequency")
i = 0
for key in sorted(d, key=d.get, reverse=True):
    if i >= 10:
        break

    print(key, d[key])
    i += 1
```

# Tuples





# What is a tuple?

- A tuple is like a list, but immutable
- Round brackets ( ) or no brackets

```
>>> t = tuple()
```

```
>>> print(t)
```

```
# comma separated:
```

```
>>> t = 'a', 'b', 'c', 'd', 'e'
```

```
# clearer with brackets:
```

```
>>> t = ('a', 'b', 'c', 'd', 'e')
```

```
>>> tup = ('a',) # single value needs end comma
```



# Tuple assignments

- All values need to be assigned at once
- It is possible to replace an entire tuple

```
>>> t = ('a', 'b', 'c', 'd', 'e')
>>> t[0] = 'bla'           # not possible
>>> t.append('bla')        # not possible
>>> t = ('bla',) + t       # possible!
                           (overwriting whole tuple)
```





# Why use tuples?

- Faster to iterate through
- “Protect” values that shouldn’t be changed
- Useful to assign multiple values at once
- Useful in combination with dictionaries



# Multiple assignments at once

```
>>> x, y = 'hello', 'there'
```

```
>>> print(x)
```

```
>>> print(y)
```

```
>>> x, y = y, x      # swap values
```

```
>>> mail = 'saschavanesse@gmail.com'
```

```
>>> name, domain = mail.split('@')
```



# Compare tuples

It starts at the first elements and stops once one is bigger/smaller

```
>>> (0,1,2) < (0,3,4) # True
```

```
>>> (0, 1, 2000000) < (0, 3, 4) # True?
```

```
>>> (10,1,2) < (0,3,4) # False
```



# Dictionaries and tuples

```
>>> d = { 'socrates': 10, 'gorgias': 5, 'polus': 2}  
>>> t = d.items()      # a list of tuples  
>>> print(t)  
>>> sorted(t) # tuple sort behaviour (previous slide)
```

## Multiple assignments in a script

```
for key, value in sorted(d.items()):  
    print(key, value)
```



# Strings, lists, dictionaries, tuples

- String: "", immutable, sequence of characters

```
>>> s = "aap"
```

- List: [], mutable, ordered, index is number

```
>>> l = ['aap', 'noot', 'mies']
```

- Dictionary: {}, mutable, unordered, key-value pairs

```
>>> d = {'aap':10, 'noot':5, 'mies':25}
```

- Tuple: (), immutable, ordered, index is number

```
>>> l = ('aap', 'noot', 'mies')
```



# Last lesson! Now what??

- Repetition!
- Learning by doing!

Next step for python:

- Based on the same book:  
<https://www.coursera.org/learn/python>
- Other options:
  - <https://www.codecademy.com/learn/learn-python>
  - <https://www.edx.org/course/introduction-to-python-absolute-beginner-0>



# Shortcuts

- Terminal

- Up      previous commands
- Tab      autocomplete
- cd ..    one folder up

- Editor

- Ctrl-/ or Cmd-/      comment on/off
- Tab                    indent forward
- Shift-Tab            indent backwards